Advanced JUnit Testing Exercises

# Exercise 1: Parameterized Tests

**EvenChecker.java**

public class EvenChecker {

public boolean isEven(int number) {

return number % 2 == 0;

}

}

**EvenCheckerTest.java**

import com.mycompany.mavenproject1.EvenChecker;

import static org.junit.jupiter.api.Assertions.\*;

import org.junit.jupiter.params.ParameterizedTest;

import org.junit.jupiter.params.provider.ValueSource;

public class EvenCheckerTest {

EvenChecker checker = new EvenChecker();

@ParameterizedTest

@ValueSource(ints = {2, 4, 6})

void testEvenNumbers(int num) {

assertTrue(checker.isEven(num), num + " should be even");

}

@ParameterizedTest

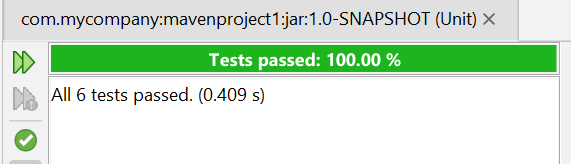
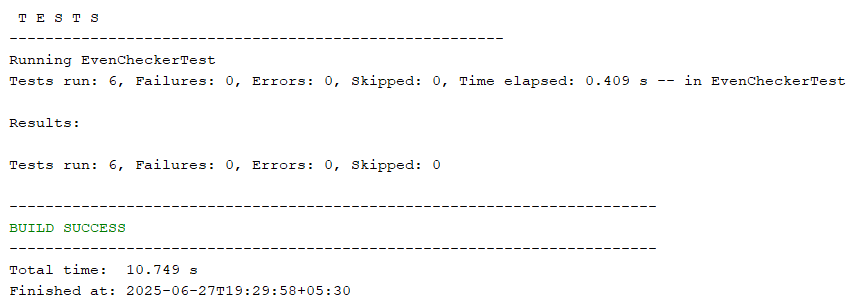
@ValueSource(ints = {1, 3, 5})

void testOddNumbers(int num) {

assertFalse(checker.isEven(num), num + " should be odd");

}

}

**Output**

# Exercise 2: Test Suites and Categories

**AllTests.java**

import org.junit.runner.RunWith;

import org.junit.runners.Suite;

@RunWith(Suite.class)

@Suite.SuiteClasses({

EvenCheckerTest.class,

ExceptionThrowerTest.class,

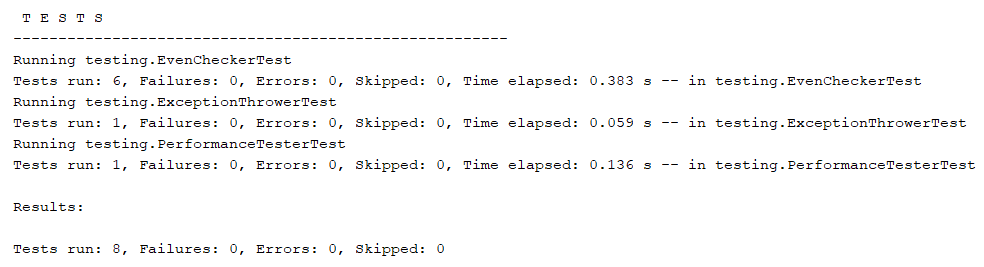
PerformanceTesterTest.class,

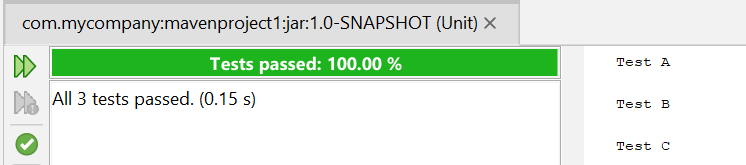
OrderedTests.class

})

public class AllTests {

}





# Exercise 3: Test Execution Order

**OrderedTests.java**

import org.junit.jupiter.api.\*;

@TestMethodOrder(MethodOrderer.OrderAnnotation.class)

public class OrderedTests {

@Test

@Order(2)

void testB() {

System.out.println("Test B");

}

@Test

@Order(1)

void testA() {

System.out.println("Test A");

}

@Test

@Order(3)

void testC() {

System.out.println("Test C");

}

}

**Output**

# 

# Exercise 4: Exception Testing

**ExceptionThrower.java**

public class ExceptionThrower {

public void throwException() {

throw new IllegalArgumentException("This is an error!");

}

}

**ExceptionThrowerTest.java**

import com.mycompany.mavenproject1.ExceptionThrower;

import static org.junit.jupiter.api.Assertions.\*;

import org.junit.jupiter.api.Test;

public class ExceptionThrowerTest {

@Test

void testExceptionThrown() {

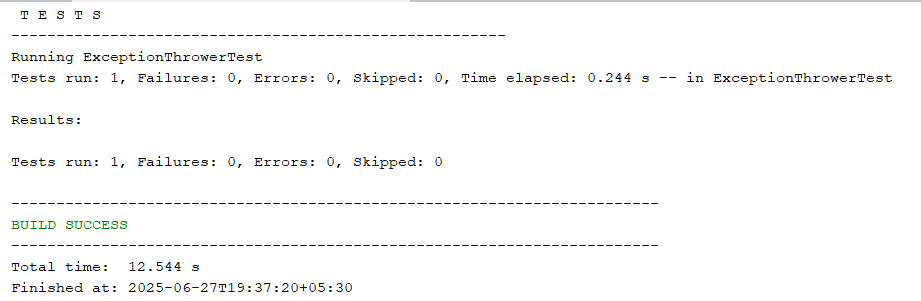
ExceptionThrower ex = new ExceptionThrower();

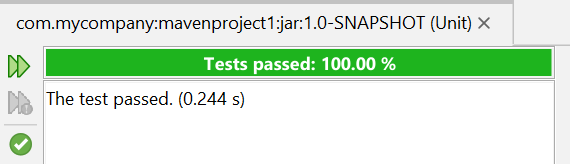
assertThrows(IllegalArgumentException.class, ex::throwException);

}

}

**Output**





# Exercise 5: Timeout and Performance Testing

**PerformanceTester.java**

public class PerformanceTester {

public void performTask() throws InterruptedException {

Thread.sleep(100);

}

}

**PerformanceTesterTest.java**

import com.mycompany.mavenproject1.PerformanceTester;

import org.junit.jupiter.api.Test;

import static org.junit.jupiter.api.Assertions.assertTimeout;

import java.time.Duration;

public class PerformanceTesterTest {

@Test

void testTaskTimeout() {

PerformanceTester pt = new PerformanceTester();

assertTimeout(Duration.ofMillis(500), () -> pt.performTask());

}

}

**Output**

